



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/845,751	04/30/2001	Bernhard J. Scholz	GE1-004US	3458
21718	7590	10/10/2006	EXAMINER	
LEE & HAYES PLLC SUITE 500 421 W RIVERSIDE SPOKANE, WA 99201			PAULA, CESAR B	
			ART UNIT	PAPER NUMBER
			2178	

DATE MAILED: 10/10/2006

Please find below and/or attached an Office communication concerning this application or proceeding.



UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

RECEIVED
OCT 10 2006
Technology Center 2100

**BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES**

Application Number: 09/845,751
Filing Date: April 30, 2001
Appellant(s): SCHOLZ ET AL.

Allan T Sponseller
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed 7/10/2006 appealing from the Office action mailed 8/8/2005.

(1) Real Party in Interest

A statement identifying by name the real party in interest is contained in the brief.

(1) Real Party in Interest

A statement identifying by name the real party in interest is contained in the brief.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims contained in the brief is correct.

(4) Status of Amendments After Final

The appellant's statement of the status of amendments after final rejection contained in the brief is correct.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal is correct.

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

Pat.# 6,292,827 Raz filed on 6/20/1997

Pat.# 6,832,369 B1 Kryka et al filed on 8/1/2000.

Art Unit: 2178

Lemay et al, "Laura Lemay's Workshop JavaScript", 1996, Sams.net, pp.132-137.

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

Specification

1. The specification is objected to as failing to provide proper antecedent basis for the claimed subject matter. See 37 CFR 1.75(d)(1) and MPEP § 608.01(o). Correction of the following is required: While the original specification refers to computer-readable instructions, it fails to provide antecedent basis for the terms "One or more computer-readable media" as used in claims 1-6 and 43. Likewise, the only mention of "data structure" or what is meant by a "portion identifying" is a field as used in claims 38-42 is in Appellant's claims.

Claim Rejections - 35 USC § 101

35 U.S.C. 101 reads as follows:

Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

2. Claims 1-6, 23-36, and 38-43 are newly rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

3. Regarding claims 1-6 and 43, such claims are directed to, "One or more computer-readable media comprising computer-executable instructions..." While the instructions are further defined in the claims, no further limitations of the one or more computer-readable media

are provided. As noted above, turning to Appellant's original disclosure and written specification provides no further guidance as to the intended metes and bounds of "One or more computer-readable media" as used in claims 1-6 and 43. While no explicit and deliberate definition of what is meant by "computer-executable instructions" is provided, Appellant's disclosure makes clear that each of the disclosed processes are "implemented as a software process of acts performed by execution of software instructions" (see, e.g., the description of process 900 contained on page 58, lines 4-8 of the specification). It should be again noted that Appellant's claims only require that the one or computer-readable media comprise these software instructions. Neither Appellant's claims themselves nor Appellant's original disclosure provide for the instructions to be stored on or recorded on the one or more computer-readable media. As such, the broadest reasonable interpretation which would be fairly conveyed to one of ordinary skill in the art in light of Appellant's original disclosure is believed to be that the "One or more computer-readable media" as used in claims 1-6 and 43 would be the "software instructions" themselves. Since "software instructions" are devoid of any physical articles or objects which may cooperate to achieve some function, such are not a machine. Likewise, absent being any such physical article or object, they cannot be a manufacture. They are clearly not a series of steps or acts themselves, and as such are not a process. They are clearly not a composition of matter. Therefore, claims 1-6 and 43 do not appear to fall within a statutory category of invention as set forth in 35 USC 101. Instead, they are believed to be directed to functional descriptive material, *per se*, and non-statutory.

Art Unit: 2178

4. Claim 23 recites a software system, which is not tangibly embodied in a manner so as to be executable. A software system is descriptive material per se and therefore not statutory subject matter. Regarding claims 23-29, such claims are directed to a system comprising two elements, a form analyzer and a tag replacement module, and claims 30-31 add a third element, a tag library. Turning again to Appellant's original disclosure, specification page 33, lines 14-19 describes an unclaimed form processor 808 which is comprised of a form analyzer module 810 and a tag replacement module 812. Page 39, lines 11-13 state, "the form processor 808 may be a separate component or module (e.g., software, firmware, and/or hardware) that analyzes the form definitions 806 to identify the custom tags." Lastly, page 66, lines 19-20 state, "[t]he discussions herein are directed primarily to software modules and components." The intrinsic evidence noted above is believed to provide reasonable basis for one of ordinary skill to interpret the systems of claims 23-29 as covering software alone. In fact, even if the currently unclaimed processor were added to the system, based on the above citations from Appellant's specification, such would not resolve the deficiency. For the reasons above, software alone is not believed to fall within a statutory category as set forth in 35 USC 101. With regard to claims 30-31, tag library 816 is described as equivalent to a database and its function is described primarily on page 34, lines 7-25 and page 52, line-6. It is unclear from Appellant's original disclosure whether the library as used in the claims is the collection of data and information or necessarily includes the underlying storage hardware. This uncertainty is further complicated by attributing the ability for functions to be "automatically built by" or "built by" the library as described in the portion of Table 4 contained on page 41. As such, it would appear reasonable for one of ordinary skill in the art to interpret the term "library" as used in claims 30-31 as software only

Art Unit: 2178

rather than necessarily including both the software and underlying hardware storage. For the reasons above, a claim directed to software only is not believed to fall within a statutory category as set forth in 35 USC 101, see MPEP 2105.

5. Regarding claims 32-36, Appellant has chosen to differentiate claim 32 from claim 19, in part, by using solely the term "adding" as opposed to "automatically adding." In doing so, each of the steps of claim 32 appear reasonably interpreted by one of ordinary skill as solely a thought. Nothing in claim 32 appears to require anything more than thoughts to take place, as opposed to requiring a practical application of the thoughts. As such, claim 32 appears to be directed purely to an abstract idea. No usefulness of having added the code to the definition can be realized from the claim as written, since the addition is believed to be reasonably interpreted by one of ordinary skill as requiring nothing more than combining two thoughts in one's mind. While claims 33-36 do not appear to cure this deficiency, claim 37 has not been included in this rejection, as the requirement to execute code brings it out of the realm of an abstract idea in the form of a thought, instead requiring a real-world act to take place that enables usefulness of the additions to be realized.

6. Claim 38 recites a data structure, which is not tangibly embodied in a computer-readable medium in a manner so as to be executable is software *per se*. Claims 38-42 are directed to what Appellant refers to as a "data structure," *per se*. Even assuming for arguments sake that Appellant's claims are, in fact, directed to a data structure, a data structure is functional descriptive material, *per se*, and non-statutory. Similar to the software instructions analysis

Art Unit: 2178

provided above, "data structures" are devoid of any physical articles or objects which may cooperate to achieve some function, and as such are not a machine. Likewise, absent being any such physical article or object, they cannot be a manufacture. They are clearly not a series of steps or acts themselves, and as such are not a process. They are clearly not a composition of matter. Therefore, like software instructions, data structures in and of themselves do not appear to fall within a statutory category of invention as set forth in 35 USC 101. Further, in this instance, only claim 39 appears to rise to the level of something one of ordinary skill in the art may reasonably consider to actually be a data structure. Page 30, lines 13-18 of Appellant's original specification describe the document as claimed in claim 39. Such would appear to constitute a data structure. From MPEP 2106.01, the "definition of 'data structure' is 'a physical or logical relationship among data elements, designed to support specific data manipulation functions.' The New IEEE Standard Dictionary of Electrical and Electronics Terms 308 (5th ed. 1993)." However, claim 39 is sufficiently vague as to render unclear whether the portions of claim 38 are part of the document of claim 39, or whether such are distinct from one another and simply part of the overall so-called data structure. One way or the other, claim 39 lacks recitation of an appropriate physical article or object that is functionally or structurally interconnected with the text markup language document in such a manner as to enable the text markup language document to act as a computer component and realize any functionality it may have. Claims 38 and 40-42 appear to be directed to (at best) an arrangement of data, non-functional descriptive material, *per se*. As written, claims 38 and 40-42 appear directed to an abstract idea in the form of a layout intended to be used for data, also see MPEP 2105.

Claim Rejections - 35 USC § 112

7. The following is a quotation of the second paragraph of 35 U.S.C. 112:

The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

8. Claims 38-42 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention. Claims 38-42 appear to be inaccurate. Appellant's use of the term "data structure" appears to be a misnomer. For the reasons above, claims 38 and 40-42 are believed to clearly be repugnant to the normal and ordinary meaning of the term "data structure," and claim 39 is sufficiently vague to likewise raise an appropriate question as to whether what is claimed is actually a "data structure" within its broadest reasonable sense. As such, claims 38-42 appear inaccurate.

Claim Rejections - 35 USC § 102

9. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

10. Claims 1, 19, and 43 remain rejected under 35 U.S.C. 102(e) as being anticipated by Raz (Pat.# 6,292,827, 9/18/2001, filed on 6/20/1997).

Regarding independent claim 1, Raz teaches the automatic conversion of paper forms into HTML-coded forms having form fields—*custom field on a source code form definition*. During the automatic conversion, validation functions are added to the converted HTML-coded form fields for verifying the data input into those fields—*identifying a custom field on a HTML source code form definition and one or more restrictions and validation code, that when executed, validates that the input conforms to the one or more restrictions* (col.12, lines 36-49). In other words, a new HTML file is formed, which comprises not only the coded fields, but also the validation functions associated with corresponding fields-- *adding to a new form definition that includes a non-custom field corresponding to the custom field, the identified validation code..*

Regarding independent claim 19, Raz teaches the automatic conversion of paper forms into HTML forms. During the automatic conversion, validation functions are added to the converted HTML-coded form fields for verifying the data input into those fields—*identifying, from an input source code definition written in a source code, one or more desired fields and automatically adding validation code to source code of the form* (col.12, lines 36-49).

Regarding claim 43, which depends on claim 1, Raz teaches the adding of validation functions to HTML-coded form fields. For example, a digit-checking validation function is added to an account number field found in the HTML form—*using, in identifying the validation code, the one or more restrictions* (col.12, lines 36-49). In other words, the form field is

evaluated to determine the type of data it will receive, in this case account digits, then a digit-checking validation form is retrieved and added to the HTML form.

Claim Rejections - 35 USC § 103

11. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

12. Claims 2-6, 20-24, and 26-42 remain rejected under 35 U.S.C. 103(a) as being unpatentable over Raz, in view of Laura Lemay's Workshop JavaScript, Lemay et al, hereinafter Lemay, 1996, Sams.net, pp.132-137.

Regarding claim 2, which depends on claim 1, Raz teaches the automatic conversion of paper forms into HTML forms. After converting the forms, validation functions—*validation code that causes the other processor to execute the identified validation code*--are added to the HTML form fields for verifying the data input into those fields (col.12, lines 36-49). Raz fails to explicitly disclose: *adding to the new form definition, a reference to the identified validation code*. However, Lemay teaches adding validation functions to HTML forms using a function call, such as "function Validate ()" page 134, line 3-- *reference to the added validation code*. The validation functions are added within HTML tags (page 133, lines 17-page 18). It would have been obvious to a person of ordinary skill in the art at the time of the invention to add the

Art Unit: 2178

validation reference to the converted form of Raz with well-known HTML function calls taught by Lemay, because Lemay teaches the saving of trouble, and receiving immediate feedback without having to wait on a server (page 132, lines 7-10). Thus, allowing a user to save time and trouble by inserting the Javascript validation functions in the created HTML tags.

Regarding claim 3, which depends on claim 1, Raz teaches the automatic conversion of paper forms into HTML forms. After converting the forms, validation functions—*pre-defined validation code* are added to the HTML form fields for verifying the data input into those fields (col.12, lines 36-49).

Regarding claim 4, which depends on claim 1, Raz teaches the automatic conversion of paper forms into HTML forms. After converting the forms, validation functions are added to the HTML form fields for verifying the data input into those fields (col.12, lines 36-49). Raz fails to explicitly disclose: *the source code form definition that defines a field includes a tag corresponding to the field*. However, Lemay teaches adding validation functions to HTML forms. The validation functions are added within HTML tags (page 133, lines 17-page 18). It would have been obvious to a person of ordinary skill in the art at the time of the invention to add the validation reference to the converted form of Raz with well-known HTML function calls taught by Lemay, because Lemay teaches the saving of trouble, and receiving immediate feedback without having to wait on a server (page 132, lines 7-10). Thus, allowing a user to save time and trouble by inserting the Javascript validation functions in the created HTML tags.

Regarding claim 5, which depends on claim 1, Raz teaches the automatic conversion of paper forms into HTML forms, having fields, such as an account field, for typing—*user input--* an account number (col.12, lines 36-49).

Regarding claim 6, which depends on claim 1, Raz teaches the automatic conversion of paper forms into HTML forms. During the automatic conversion, validation functions are added to the converted form fields for verifying the data input into those fields—*identifying, from a plurality of pieces of validation code, the validation code corresponding to the one or more attributes of the custom tag* (col.12, lines 36-49). Raz fails to explicitly disclose: *identifying a custom tag corresponding to the field, wherein the custom tag includes an indication of one or more attributes and wherein each of the one or more attributes includes a value indicating what input corresponding to the non-custom field is to be restricted to*. However, Lemay teaches adding validation functions to HTML forms. The validation functions are added within HTML tags, and the functions indicate an attribute, such as “ValidLength”, and length value not to be exceeded (page 133, lines 17-page 18). It would have been obvious to a person of ordinary skill in the art at the time of the invention to include the validation functions of Raz within tags of Lemay, because Lemay teaches the saving of trouble, and receiving immediate feedback without having to wait on a server (page 132, lines 7-10). Thus, allowing a user to save time and trouble by inserting the Javascript validation functions in the created HTML tags—*custom tags*.

Regarding claim 20, which depends on claim 19, Raz teaches the automatic conversion of paper forms into HTML forms. During the automatic conversion, validation functions are added

Art Unit: 2178

to the converted form fields for verifying the data input into those fields (col.12, lines 36-49). Raz fails to explicitly disclose: *a custom tag corresponding to each of the one or more desired fields, wherein each custom tag has one or more validation attributes and wherein each validation attribute includes an indication of the attribute and a corresponding value that input corresponding to the custom tag is to be restricted to.* However, Lemay teaches adding validation functions to HTML forms. The validation functions are added within HTML tags, and the functions indicate an attribute, such as “ValidLength”, and length value not to be exceeded (page 133, lines 17-page 18). It would have been obvious to a person of ordinary skill in the art at the time of the invention to include the validation functions of Raz within tags of Lemay, because Lemay teaches the saving of trouble, and receiving immediate feedback without having to wait on a server (page 132, lines 7-10). Thus, allowing a user to save time and trouble by inserting the Javascript validation functions in the created HTML tags—*custom tags*.

Regarding claim 21, which depends on claim 19, Raz teaches the automatic conversion of paper forms into HTML forms. During the automatic conversion, validation functions are added to the converted form fields for verifying the data input by a user into those fields (col.12, lines 36-49).

Regarding claim 22, which depends on claim 19, Raz teaches the automatic conversion of paper forms into HTML forms—*generating a temporary form definition*. During the automatic conversion, validation functions are added to the converted form fields for verifying the data input into those fields—*adding and executing code to add the identified validation code to the*

Art Unit: 2178

new form definition and outputting, as the source code, the temporary form definition (col.12, lines 36-49).

Régarding independent claim 23, Raz teaches the automatic conversion of paper forms into HTML forms. After converting the forms, validation functions are added to the HTML form fields for verifying the data input into those fields — *further to add to a form definition, for each of the one or more custom tags, validation code to validate subsequent inputs to a field* (col.12, lines 36-49). Raz fails to explicitly disclose: *one or more custom tags in a source code form definition; and replace each of the one or more custom tags with another tag, and field corresponding to the tag*. However, Lemay teaches adding validation functions to HTML forms. The validation functions are added within HTML tags in a form file, and the functions indicate an attribute, such as “ValidLength”, and length value not to be exceeded (page 133, lines 17- page 18). It would have been obvious to a person of ordinary skill in the art at the time of the invention to include the validation functions into the HTML form of Raz within tags as suggested by Lemay, because Lemay teaches the saving of trouble, and receiving immediate feedback without having to wait on a server (page 132, lines 7-10). Thus, allowing a user to save time and trouble by inserting the Javascript validation functions in the created HTML tags— *custom tags*.

Regarding claim 24, which depends on claim 23, Raz teaches the automatic conversion of paper forms into HTML forms. During the automatic conversion, validation functions are added

Art Unit: 2178

to the converted form fields for verifying the data input by a user into those fields (col.12, lines 36-49).

Regarding claim 26, which depends on claim 23, Raz teaches the automatic conversion of paper forms into HTML forms. After converting the forms, validation functions are added to the HTML form fields for verifying the data input into those fields (col.12, lines 36-49). Raz fails to explicitly disclose: *a custom tag is a HTML tag*. However, Lemay teaches adding validation functions to HTML forms. The validation functions are added within HTML tags (page 133, lines 17-page 18). It would have been obvious to a person of ordinary skill in the art at the time of the invention to replace the tags of the converted form of Raz with well-known HTML tags suggested by Lemay, because Lemay teaches the saving of trouble, and receiving immediate feedback without having to wait on a server (page 132, lines 7-10). Thus, allowing a user to save time and trouble by inserting the Javascript validation functions in the created HTML tags—*custom tags*.

Regarding claim 27, which depends on claim 23, Raz teaches the automatic conversion of paper forms into HTML forms. After converting the forms, validation functions are added to the HTML form fields for verifying the data input into those fields (col.12, lines 36-49). Raz fails to explicitly disclose: *add a reference to the added validation code*. However, Lemay teaches adding validation functions to HTML forms using a function call, such as “function Validate ()” page 134, line 3-- *reference to the added validation code*. The validation functions are added within HTML tags (page 133, lines 17-page 18). It would have been obvious to a person of ordinary skill in the art at the time of the invention to add the validation reference to the

Art Unit: 2178

converted form of Raz with well-known HTML function calls taught by Lemay, because Lemay teaches the saving of trouble, and receiving immediate feedback without having to wait on a server (page 132, lines 7-10). Thus, allowing a user to save time and trouble by inserting the Javascript validation functions in the created HTML tags.

Regarding claim 28, which depends on claim 23, Raz teaches the automatic conversion of paper forms into HTML forms—*form definition*. After converting the forms, validation functions are added to the HTML form fields-- *generate a new document corresponding to the form definition*-- for verifying the data input into those fields-- *add validation code to the new document* (col.12, lines 36-49). Raz fails to explicitly disclose: *to replace each of the one or more custom tags with another tag by adding the other tag to the new document*. However, Lemay teaches adding validation functions to HTML forms. The validation functions are added within HTML tags (page 133, lines 17-page 18). It would have been obvious to a person of ordinary skill in the art at the time of the invention to add the validation code to the newly converted form of Raz within well-known HTML tags taught by Lemay, because Lemay teaches the saving of trouble, and receiving immediate feedback without having to wait on a server (page 132, lines 7-10). Thus, allowing a user to save time and trouble by inserting the Javascript validation functions in the created HTML tags.

Regarding claim 29, which depends on claim 23, Raz teaches the automatic conversion of paper forms into HTML forms. After converting the forms, validation functions are added to the HTML form fields for verifying the data input into those fields—*restrictions corresponding to*

Art Unit: 2178

the same validation code, add the same validation code only once (col.12, lines 36-49). Raz fails to explicitly disclose: *one or more custom tags*. However, Lemay teaches adding validation functions to HTML forms. The validation functions are added within HTML tags (page 133, lines 17-page 18). It would have been obvious to a person of ordinary skill in the art at the time of the invention to add the validation code to the newly converted form of Raz within well-known HTML tags taught by Lemay, because Lemay teaches the saving of trouble, and receiving immediate feedback without having to wait on a server (page 132, lines 7-10). Thus, allowing a user to save time and trouble by inserting the Javascript validation functions in the created HTML tags.

Regarding claim 30, which depends on claim 23, Raz teaches the automatic conversion of paper forms into HTML forms. After converting the forms, validation functions are added to the HTML form fields for verifying the data input into those fields. The data, including validation functions, are stored in a RDBMS database—*tag library* (col. 9, lines 27-45, col.12, lines 36-49).

Regarding claim 31, which depends on claim 30, Raz teaches the automatic conversion of paper forms into HTML forms. After converting the forms, validation functions are added to the HTML form fields for verifying the data input into those fields. The data, including validation functions, are stored in a RDBMS database—*tag library* (col. 9, lines 27-45, col.12, lines 36-49). Raz fails to explicitly disclose: *an identification of the one or more custom tags*. However, Lemay teaches adding validation functions to HTML forms. The validation functions are added within HTML tags--*identification* (page 133, lines 17-page 18). It would have been obvious to a

Art Unit: 2178

person of ordinary skill in the art at the time of the invention to combine Raz, and Lemay to store the well-known HTML tags taught by Lemay, because Lemay teaches the saving of trouble, and receiving immediate feedback without having to wait on a server (page 132, lines 7-10). Thus, allowing a user to save time and trouble by inserting the Javascript validation functions in the created HTML tags.

Regarding independent claim 32, Raz teaches the automatic conversion of paper forms into HTML forms—*form definition*(col.12, lines 36-49).

Moreover, Raz teaches that after converting the forms, validation functions are added to the HTML form fields for verifying the data input into those fields — *identifying and adding validation code that, when executed based on an input corresponding to the field, validates whether the associated restrictions are satisfied* (col.12, lines 36-49). In other words, the converted HTML-coded form(s) is received, and then validation functions associated—*one or more associated input restrictions--* with the HTML form fields, are added to the form. Raz fails to explicitly disclose: *identifying a replacement non-custom tag, adding the identified replacement non-custom tag to a new form definition*. However, Lemay teaches adding validation functions to HTML forms along with tags— *identifying and adding a replacement non-custom tag*. The validation functions are added within HTML tags, and the functions indicate an attribute, such as “ValidLength”, and length value not to be exceeded (page 133, lines 17-page 18). It would have been obvious to a person of ordinary skill in the art at the time of the invention to have added the validation tags of Lemay and functions to the form fields of Raz, because Lemay teaches the saving of trouble, and receiving immediate feedback without having

Art Unit: 2178

to wait on a server (page 132, lines 7-10). Thus, allowing a user to save time and trouble by inserting the Javascript validation functions in the created HTML tags—*custom tags*.

Regarding claim 33, which depends on claim 32, Raz teaches the automatic conversion of paper forms into HTML forms. After converting the forms, validation functions are added to the HTML form fields for verifying the data input into those fields (col.12, lines 36-49). Raz fails to explicitly disclose: *adding to the new form definition, a reference to invoke the added validation code*. However, Lemay teaches adding validation functions to HTML forms using a function call, such as “function Validate ()” page 134, line 3-- *reference to the added validation code*. The validation functions are added within HTML tags (page 133, lines 17-page 18). It would have been obvious to a person of ordinary skill in the art at the time of the invention to add the validation reference to the converted form of Raz with well-known HTML function calls taught by Lemay, because Lemay teaches the saving of trouble, and receiving immediate feedback without having to wait on a server (page 132, lines 7-10). Thus, allowing a user to save time and trouble by inserting the Javascript validation functions in the created HTML tags.

Regarding claim 34, which depends on claim 32, Raz teaches that after converting the forms—*receiving HTML form definition*, validation functions are added to the HTML form fields for verifying the data input into those fields (col.12, lines 36-49). Raz fails to explicitly disclose: *adding the each of the non-custom tags to the new form definition*. However, Lemay teaches adding validation functions to HTML forms along with tags— *identifying and adding a replacement non-custom tag*. The validation functions are added within HTML tags, and the

functions indicate an attribute, such as “ValidLength”, and length value not to be exceeded (page 133, lines 17-page 18). It would have been obvious to a person of ordinary skill in the art at the time of the invention to have added the validation tags of Lemay and functions to the form fields of Raz, because Lemay teaches the saving of trouble, and receiving immediate feedback without having to wait on a server (page 132, lines 7-10). Thus, allowing a user to save time and trouble by inserting the Javascript validation functions in the created HTML tags—*custom tags*.

Regarding claim 35, which depends on claim 32, Raz teaches the automatic conversion of paper forms into HTML forms. During the automatic conversion, validation functions are added to the converted form fields for verifying the data input by a user into those fields (col.12, lines 36-49).

Regarding claim 36, which depends on claim 32, Raz teaches the automatic conversion of paper forms into HTML forms. During the automatic conversion, validation functions are added to the converted form fields for verifying the data input into those fields (col.12, lines 36-49). Raz fails to explicitly disclose: *each input custom tag includes one or more attributes that identify the one or more associated restrictions, and wherein each of the one or more attributes includes an indication of the attribute and a corresponding value for that data input corresponding to the tag is to be restricted to*. However, Lemay teaches adding validation functions to HTML forms. The validation functions are added within HTML tags, and the functions indicate an attribute, such as “ValidLength”, and length value not to be exceeded (page 133, lines 17-page 18). It would have been obvious to a person of ordinary skill in the art at the

Art Unit: 2178

time of the invention to include the validation functions of Raz within attributes of Lemay, because Lemay teaches the saving of trouble, and receiving immediate feedback without having to wait on a server (page 132, lines 7-10). Thus, allowing a user to save time and trouble by inserting the Javascript validation functions in the created HTML tags.

Regarding claim 37, which depends on claim 32, Raz teaches the automatic conversion of paper forms into HTML forms. During the automatic conversion, validation functions are added to the converted form fields for verifying the data input into those fields—*execution code to add the identified validation code to the new form definition* (col.12, lines 36-49).

Regarding independent claim 38, Raz teaches the automatic conversion of paper forms into HTML forms, having fields for entering data. After converting the forms, validation functions are added to the HTML form fields—a *first portion identifying an input field--* for verifying the data input into those fields — *a second portion identifying validation code to be added to a page to enforce the one or more restrictions* (col.12, lines 36-49).

Regarding claim 39, which depends on claim 30, Raz teaches the automatic conversion of paper forms into HTML forms—*text markup language document*. After converting the forms, validation functions are added to the HTML form fields for verifying the data input into those fields (col.12, lines 36-49).

Regarding claim 40, which depends on claim 38, Raz teaches the automatic conversion of paper forms into HTML forms, having fields, such as an account field, for typing an account number (col.12, lines 36-49).

Regarding claim 41, which depends on claim 38, Raz teaches the automatic conversion of paper forms into HTML forms. During the automatic conversion, validation functions are added to the converted form fields for verifying the data input into those fields (col.12, lines 36-49). Raz fails to explicitly disclose: *one or more attributes and for each attribute and associated value for the attribute*. However, Lemay teaches adding validation functions to HTML forms. The validation functions are added within HTML tags, and the functions indicate an attribute, such as “ValidLength”, and length value not to be exceeded (page 133, lines 17-page 18). It would have been obvious to a person of ordinary skill in the art at the time of the invention to include the validation functions of Raz within attributes of Lemay, because Lemay teaches the saving of trouble, and receiving immediate feedback without having to wait on a server (page 132, lines 7-10). Thus, allowing a user to save time and trouble by inserting the Javascript validation functions in the created HTML tags.

Regarding claim 42, which depends on claim 38, Raz teaches the automatic conversion of paper forms into HTML forms, having fields, such as an account field, for typing—*user input*--an account number (col.12, lines 36-49).

13. Claim 25 remains rejected under 35 U.S.C. 103(a) as being unpatentable over Raz in view of Lemay, and further in view of Kryka et al, hereinafter Kryka (US Pat.# 6,832,369 B1, 12/14/2004, filed on 8/1/2000).

Regarding claim 25, which depends on claim 23, Raz teaches the automatic conversion of paper forms into HTML forms. After converting the forms, validation functions are added to the HTML form fields for verifying the data input into those fields (col.12, lines 36-49). Raz fails to explicitly disclose: *the system comprises a compiler*. However, Kryka teaches a Java compiler for compiling source code into Java bytecode form (col.1, lines 54-68). It would have been obvious to a person of ordinary skill in the art at the time of the invention to have included a compiler, because Kryka teaches above the popular use of a platform-independent programming language--Java. Thus, allowing a user to easily implement the form across multiple computer platforms.

(10) Response to Argument

Regarding claim 1, the Appellants indicate that Raz does not disclose or suggest the identification of a custom field on a source code form definition, and one of more restrictions to an input to the custom field (page 7). The Examiner disagrees, because Raz teaches the conversion of paper forms into HTML documents--source code. **Validation functions are added to the HTML code** for validating and checking input made into the fields that are in the HTML form (col.12, lines 36-49). **In other words, the fields are selected, and then the validation functions are identified and added to the HTML,** for checking the validity of the input made to the specific selected field(s), such as verifying that an account field only the appropriate digits

Art Unit: 2178

input into it. The fields, which have the selected validation functions added, are HTML code fields, and not the fields on the paper form as purported by the Appellants.

Regarding claim 19, the Appellants indicate that in order for Raz to identify the input form definition written in a source code, the forms described by Raz would have to be in a written source code definition (page 8). This is exactly what's being shown by Raz's selection of the HTML code form fields in order to add the validation functions to those fields (col.12, lines 36-49). **One has to keep in mind that the paper form is converted into HTML code, and it is that code that is used for adding the validation functions, and not the paper forms.**

Regarding claim 43, the Appellants appears to misread what Raz is teaching. The Appellant note that Raz would have to show the identification on the paper-based form (page 9, parag.2). As indicated above, the teachings of Raz are directed towards the selection of HTML code form fields, after the paper form has been converted to HTML, for adding the functions to the fields.

Claims 2-6, and 20-24, and 26-31 are rejected at least based on their dependency on claims 1, 19, and the teachings of Raz as explained above.

Regarding claims 32-35, and 37, the Appellants point out that in order for Raz to receive a form definition including one or more custom tags, the paper forms described by Raz would have to include the one or more custom tags, and the restrictions included in the custom tags

Art Unit: 2178

(page 13). The form relied upon by the rejection is the HTML form to which validation functions are added, which is made up of HTML tags describing the various fields, and formatting of the form as it was well-known in the art, and as described by Lemay (page 133).

Regarding claim 36, the Appellants submit that neither Raz, nor Lemay teach or suggest custom tag that include attribute corresponding to one or more restrictions (page 14). The Examiner disagrees, because Lemay teaches the addition of validation functions to HTML forms. The attributes identifying the functions are textual names included within HTML tags in the form (page 133). It would have been obvious to one of ordinary skill in the art to combine the addition of validating functions to an HTML form as taught by Raz, and the addition of Javascript functions to HTML tags as shown by Lemay, because of all the reasons found in Lemay, not excluding allowing the HTML form to conduct the validation, thus saving the time and trouble of having to rely on a server to perform such operations (page 132, lines 7-10).

Regarding claims 38-42, the Appellants point out that Raz does not teach a structure, which includes a first portion identifying an input field, and a second portion identifying restrictions to the fields (page 16). The Examiner disagrees, because Raz teaches that the HTML form—*data structure*-- not only includes form input fields—*first portion*, but also functions for validating or restricting—*second portion*-- input to those fields (col.12, lines 36-49).

Claim 25 is rejected at least based on their dependency on claim 23, and the teachings of Raz as explained above.

For the above reasons, it is believed that the rejections should be sustained.

(11) Related Proceeding(s) Appendix

No decision rendered by a court or the Board is identified by the examiner in the Related Appeals and Interferences section of this examiner's answer.

(11) Related Proceeding(s) Appendix

No decision rendered by a court or the Board is identified by the examiner in the Related Appeals and Interferences section of this examiner's answer.

This examiner's answer contains a new ground of rejection set forth in section **(9.1-8)** above. Accordingly, appellant must within **TWO MONTHS** from the date of this answer exercise one of the following two options to avoid *sua sponte* **dismissal of the appeal** as to the claims subject to the new ground of rejection:

(1) Reopen prosecution. Request that prosecution be reopened before the primary examiner by filing a reply under 37 CFR 1.111 with or without amendment, affidavit or other evidence. Any amendment, affidavit or other evidence must be relevant to the new grounds of rejection. A request that complies with 37 CFR 41.39(b)(1) will be entered and considered. Any request that prosecution be reopened will be treated as a request to withdraw the appeal.

(2) Maintain appeal. Request that the appeal be maintained by filing a reply brief as set forth in 37 CFR 41.41. Such a reply brief must address each new ground of rejection as set forth in 37 CFR 41.37(c)(1)(vii) and should be in compliance with the other requirements of 37 CFR

Art Unit: 2178

41.37(c). If a reply brief filed pursuant to 37 CFR 41.39(b)(2) is accompanied by any amendment, affidavit or other evidence, it shall be treated as a request that prosecution be reopened before the primary examiner under 37 CFR 41.39(b)(1).

Extensions of time under 37 CFR 1.136(a) are not applicable to the TWO MONTH time period set forth above. See 37 CFR 1.136(b) for extensions of time to reply for patent applications and 37 CFR 1.550(c) for extensions of time to reply for ex parte reexamination proceedings.

Respectfully submitted,

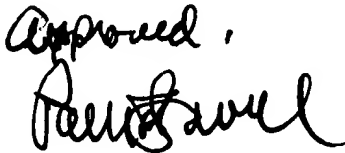


CESAR PAULA
PRIMARY EXAMINER

Cesar B Paula

October 2, 2006

A Technology Center Director or designee must personally approve the new ground(s) of rejection set forth in section (9) above by signing below:



PAUL SEWELL
ACTING DIRECTOR

Art Unit: 2178

Conferees:

A handwritten signature in black ink, appearing to read "Heather Herndon", with a stylized flourish at the end.

Heather Herndon

SPE, AU 2176

A handwritten signature in black ink, appearing to read "Stephen Hong", with a stylized flourish at the end.

Stephen Hong

SPE, AU 2178